Agentic RAG Chatbot for Multi-Format Document QA

- Powered by Model Context Protocol (MCP)
- Author: RAGUNATH R
- Date: July 2025

Problem Statement & Objective

- Problem: Build a chatbot to answer questions from diverse document formats
- Goal: Enable intelligent document Q&A with modular agents and message-driven architecture
- Supported Formats: PDF, PPTX, CSV, DOCX, TXT/Markdown

Architecture Overview (Agentic + MCP)

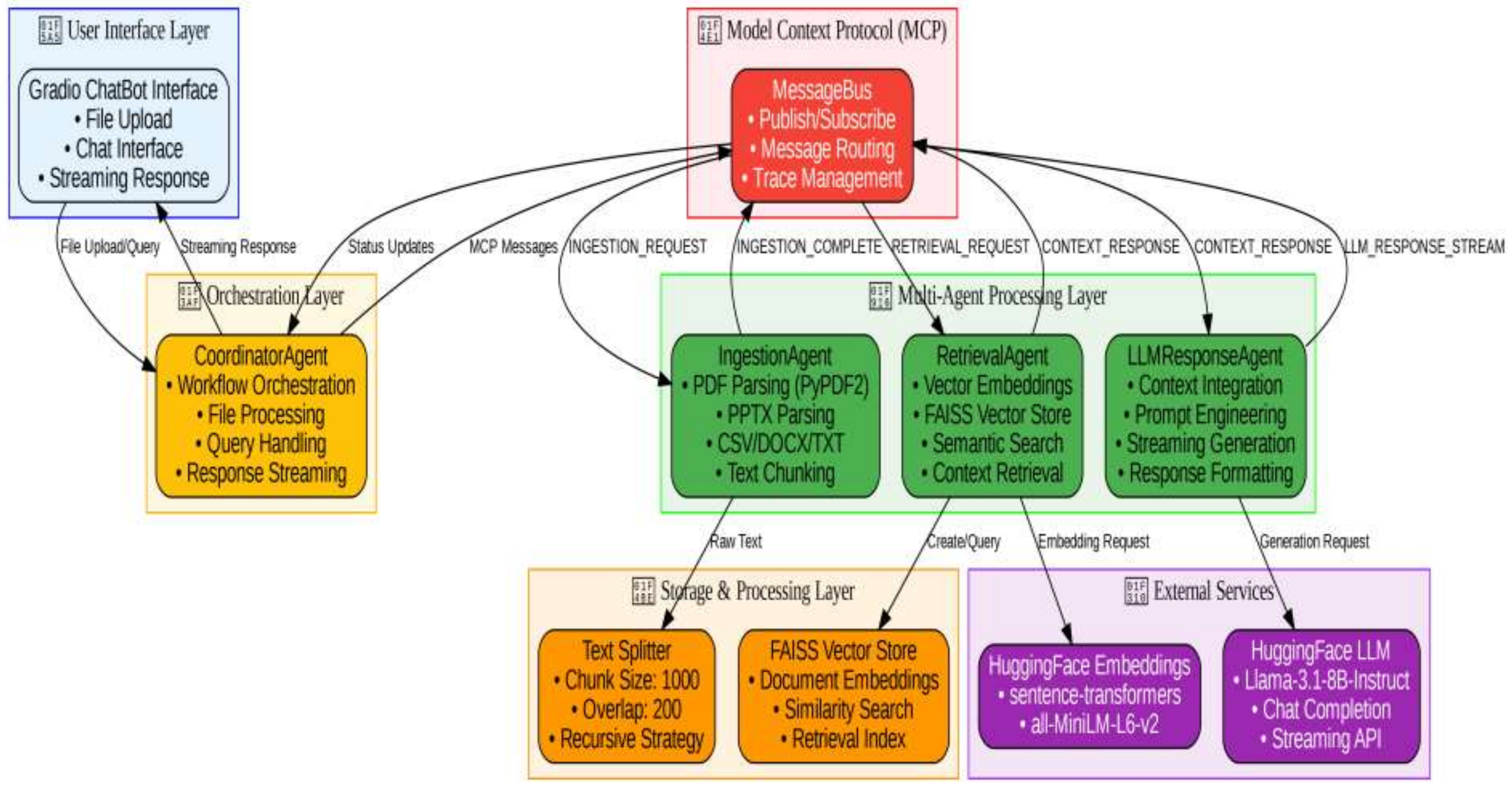
Agents:

- IngestionAgent: Parses and splits documents
- RetrievalAgent: Embeds and searches chunks
- LLMResponseAgent: Generates answers from context
- CoordinatorAgent: Manages flow & tracks context

MCP Messaging:

- Structured messages for communication between agents
- Fields: sender, receiver, type, trace_id, payload

Architecture Overview (Agentic + MCP)



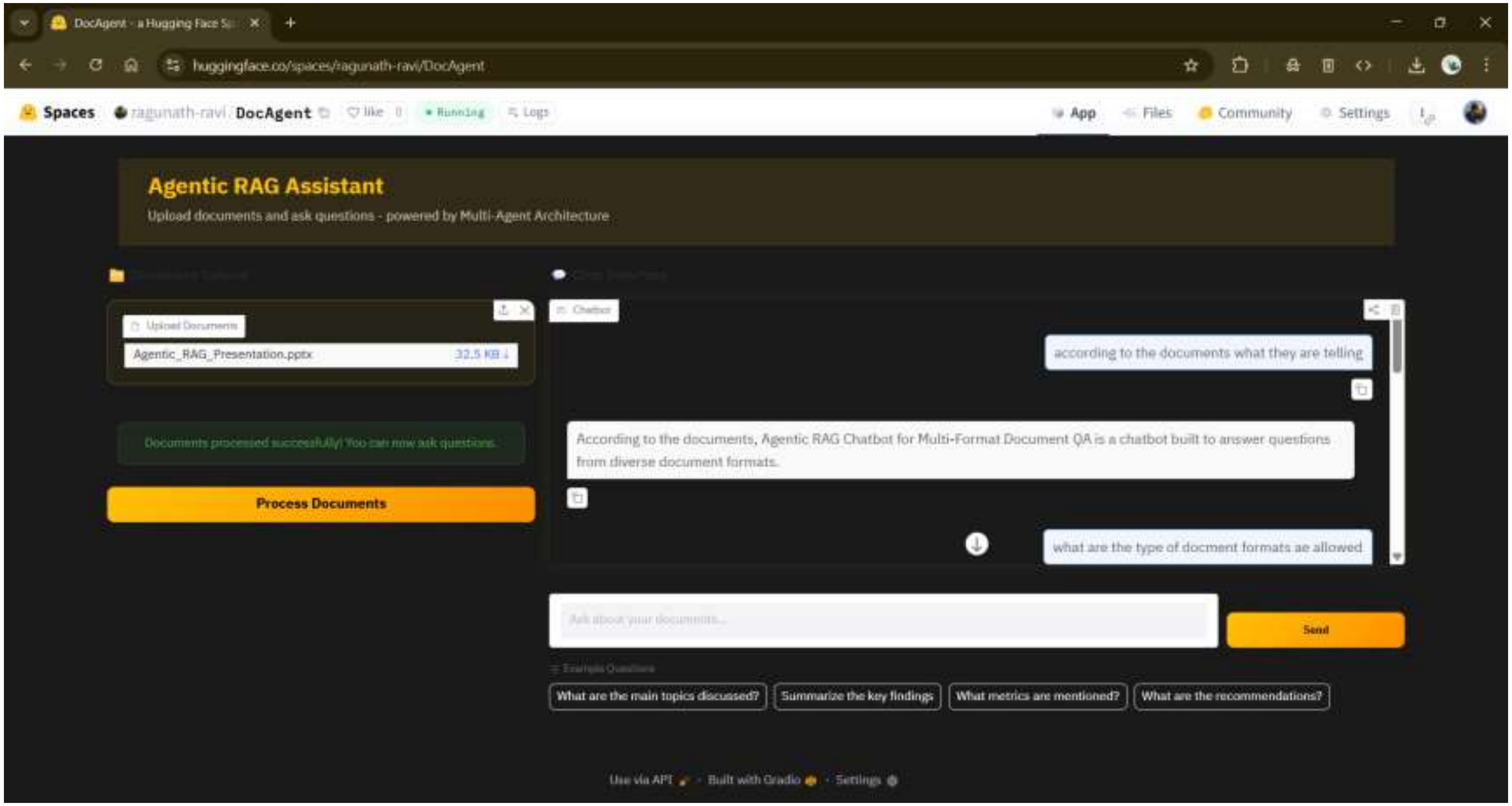
System Flow with MCP Message Passing

- 1. Upload → Coordinator: INGESTION_REQUEST
- 2. Coordinator → IngestionAgent → RetrievalAgent → LLMResponseAgent
- 3. Response streamed back to UI with sources
- MCP Message: {type, sender, receiver, trace_id, payload}

Tech Stack Used

- Frontend: Gradio (with custom CSS)
- LLM: Llama 3.1 8B via HuggingFace Infernce cli
- Vector Store: FAISS
- Embeddings: sentence-transformers/all-MiniLM-L6-v2
- Parsers: PyPDF2, python-docx, python-pptx, pandas
- Messaging: In-memory bus using Python classes

DocAgent-Demo



Agentic RAG Assistant

Upload documents and ask questions - powered by Multi-Agent Architecture

Agentic_RAG_Presentation.pptx 32.5 KB

Documents processed successfully! You can now ask questions.

according to the documents what they are telling

According to the documents, Agentic RAG Chatbot for Multi-Format Document QA is a chatbot built to answer questions from diverse document formats.

what are the type of docment formats ae allowed

Ask about your documents...

What are the main topics discussed? | Summarize the key findings | What metrics are mentioned? | What are the recommendations?

UI Features & Challenges

- Multi-file upload, multi-turn chat, visible examples
- Dark theme UI with styled messages and input box
- Challenges: LLM streaming consistency, parser errors, message delays
- Future: REST-based MCP, memory agent, GPU optimization